CHAPTER 19

BY:

ETISHA JAIN

VISHU GOYAL

Linux 19.1 Understanding Bash shell scripts

. A shell script can be as simple as a number of commands that is sequentially executed.
. Scripts normally work with variables to make them react differently in different environments.
. Conditional statements such as for,if,case and while can be used
. Shell scripts are common as they are easy to learn and implement
. Also , a shell will always be available to interpret code from shell scripts (sh)
. If the scripts use internal commands only, they are very fast as noting need to be loaded.
. There is no need to compile anything.
. There are no modules to be used in the bash script which makes them rather static .
. Also , Bash shell script are not idempotent(running same script more than once you will get same result) .

Linux 19.2 Essential Shell scripts Components

- ls -l myscript
- ./myscript
- chmod +x myscript
- ./myscript
- myscript
- echo $PATH
- vim myscript

c-bang on first line (it is identifying the command interpreter that used for running thus script. It should always start with hash exclamation mark followed by bin bash means it will always be interpreted by bash shell)

Only on first line if it starts with #! it's is a c-bang on every other line it's interpreted as comment that means it's not interpreted at all.

- read means the script is going to read user input upto moment when user presses enter and it is stored in a variable.
- $DIR takes the current value of variable
- ./myscript
- when script is over we are back in /root

It is becoz script is starting a subshell (like a new shell) and everything is done it in the new shell environment(isolated environment)and once it is done it exits upto parent shell. If you want to keep it after shell completes then we need to source the script .

Sourcing the script means that the script is not executed as a subshell but it is included in the current shell.

We can do it in two ways-

. Run it as source space name of script.

. Run as dot space name of the script.

- . myscript

Linux 19.3 Using loops in shell scripts

BASH SHELL CONDITIONAL STATEMENTS

. Different Conditional statements are available in bash
. if ... then ... fi
. while ... do ... done
. until ... do ... done
. case ... in ... e
. for ... in ... do ...done

Practical

- ls -l myargument
- chmod +x myargument
- vim myargument

Test command can be written in two ways

. test -z $1
. [-z $1]

- man test
. $1 is the first argument (something that we put behind a script)
. -z $1 will check if we have forgotten to provide an argument.
. Exit is going to stop the script immediately . We can specify exit code if not specified we have exit0(means script was successful) and exit1 (means not successful)
- ./myargument
- echo $? (To see the exit code which run in the last command)
- ./myargument abc

Linux 19.4 Using loops in Shell Scripts

- vim countdown
- echo $((2 + 4)) - - - To do calculations
- echo $((5 / 2)) - - - Not run on fraction just print 2
- && means if the test command is true run the first command otherwise the second command
- #- is construction here . It is clear pattern matching . The pattern matching operator is going to change the current value of variable counter . Hash minus is removing the minus from minus two
- chmod +x countdown
- ./countdown 1

THANK YOU